Indonesian Journal of Electrical Engineering and Computer Science
Vol. 15, No. 2, August 2019, pp. 1109~1118
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v15.i2.pp1109-1118

Constructing population of initial university timetable: design and analysis

Juliana Wahid[1], Syariza Abdul-Rahman[2], Aniza Mohamed Din[3], Naimah Mohd-Hussin[4]
[1,3]School of Computing, Universiti Utara Malaysia, Malaysia
[2]Institute of Strategic Industrial Decision Modeling (ISIDM), Decision Science Department, School of Quantitative Sciences, Universiti Utara Malaysia, Malaysia
[4]Faculty of Computer Science and Mathematical, Universiti Teknologi MARA (Perlis), Malaysia

Article Info

Article history:

Received Sep 1, 2018
Revised Feb 10, 2018
Accepted Feb 25, 2019

Keywords:

Curriculum based university course timetabling
Graph heuristics
Initial solution
Population based metaheuristic
Statistical analysis

ABSTRACT

The construction of population of initial timetable is an essential stage in population-based metaheuristic approach for solving curriculum-based university course timetabling problem because it may impact the quality of the final timetable. This paper presents population of initial timetable construction approach in curriculum based course timetabling problem by using the graph heuristics to determine the sequential order of courses/lectures to be assigned in the timetable. The graph heuristics were implemented as single and combination of two heuristics. The courses in curriculum-based university course timetabling problem that was organized based on the heuristics setting will be repeatedly assigned to valid empty slots while fulfilling all the hard constraints. If a course is unable to be assigned to whichever slots because of no more valid empty slots, it will be inserted into the unscheduled courses/lectures list. The unscheduled courses/lectures list will be assigned later to the timetable using several procedures executed in a sequence. The approaches were tested on the ITC2007 instances and the results were analyzed with some statistical tests to determine the best setting of heuristics in the construction approach. The result shows that the construction approach with combination of largest degree followed by saturation degree heuristic, generate the maximum number of population of initial timetables. The result from this study can be used in the improvement stage of metaheuristic algorithm that uses population-based approach.

Corresponding Author:

Juliana Wahid,
School of Computing,
Universiti Utara Malaysia,
06010 Sintok, Sintok Kedah.
Email: w.juliana@uum.edu.my

1. INTRODUCTION

The curriculum-based university course timetabling (CBCTT) which is produced based on university curricula, assign the lectures (came from courses) to suitable classrooms and timeslots. The allocation is established based on a set of constraints and designated on a weekly basis. The entities involved in CBCTT includes *Days*, *Timeslots*, *Periods*, *Rooms*, *Curricula*, *Courses* and *Teachers* [1]. *Days* are quantity of teaching days in a week. *Timeslots* are amount of slots in a day. *Periods* are amount of combination between *Days* and *Timeslots*. *Rooms* are halls with number of seats. *Curricula* is a set of courses in which any pair of courses in the set have students in common. *Courses* are consists of number of lectures and each lecture is associated to a *Teacher*. The conflict between courses in CBCTT are produced based on the *Curricula*.

Allocating the lectures to classrooms and periods is determined on satisfying several constraints, so that a feasible timetable can be produced. A feasible timetable is a timetable with no hard constraint violation, but perhaps contain soft constraint violation. The soft constraint violation is showed by amount of penalty cost. Therefore, solving CBCTT problem basically is to decrease that amount of penalty cost in a feasible timetable. This study adopts the hard and soft constraints used in [2]. The hard constraints are lectures, room occupancy, conflicts and availability, while the soft constraints are room capacity, minimum working days, curriculum compactness and room stability. The detail of each constraint is described in Table 1. The quality of solution is calculated as the total penalties of the soft constraints: room capacity + minimum working days + curriculum compactness + room stability.

Table 1. Description of hard and soft constraints for CBCTT

Type	Constraints	Description
Hard constraints	Lectures	All lectures must be scheduled to a different period and a room
	Room occupancy	Two lectures cannot be assigned to the same room and the same period
	Conflicts	Lectures in the same curriculum or taught by the same teacher must be assigned to separate periods
	Availability	If the teacher of a course is not available at a certain period, then no lectures of the course can be allocated to that period
Soft constraints	Room capacity	The quantity of students for each lecture must be fewer or equivalent to the volume of the rooms
	Minimum working days	The lectures of each course should span thru a given number of days
	Curriculum compactness	Lectures of courses of the same curriculum should be in contiguous periods
	Room stability	All lectures of a specific course are supposed to be allocated to the same room

The stages in the process of solving the CBCTT problem usually comprise of the construction and the improvement phases [3]. In the construction stage, an empty timetable will be progressively inserted by a lecture one by one in repetition and at the same time the violation of the hard constraint is also inspected. At the end of this stage, an initial timetable with no hard constraint violation will be produced. However, the initial timetable may have many soft constraint violation. In the improvement stage, the soft constraint violation in the initial timetable will be iteratively decreased to a minimum or zero value. The aim of this stage is to produce a better and reliable timetable.

For the construction stage, the sequential constructive algorithm is widely studied and employed in university timetabling problem [4-10]. Sequential constructive algorithm uses a strategy to select the next event and construct the full timetable by assigning the events one at a time (sequentially) to a selected timeslot. The strategy is to order the events that are not yet scheduled according to the difficulties in scheduling them into a feasible timeslot (without violating any hard constraints) [11]. A number of commonly used strategies have been adopted from the graph coloring problem [11] such as least saturation degree (SD) - in each step of the timetable construction, an event which has the least number of available slots in the timetable constructed so far is selected, largest degree (LD) - events are ordered in descending manner by the number of conflicts they have with other events, largest weighted degree (LW) - events are ordered decreasing by the total number of students in conflict with other event, largest enrolment (LE) - events are ordered decreasing by the total number of student enrolments, largest color degree (CD) - events are ordered decreasing in terms of the other events that are in conflict and have already been scheduled in the timetable, and random ordering (RO) - events that are not yet scheduled are ordered randomly.

In the context of the CBCTT benchmark data sets, the graph coloring based heuristic orderings has been implemented mostly using SD [12-15] and LD [16, 17]. It was observed that the use of the graph heuristic alone in single phase led to infeasible timetable for some problem instances [15].

For the improvement stage, the improvement process is an iterative process in which the initial timetable is modified at each step in order to minimize the soft constraint violation in the timetable. The most common approach for the iterative improvement is metaheuristic methods [18]. There are two types of methods in metaheuristic such as local search-based methods and population-based methods [18]. Local search-based methods used in solving CBCTT include simulated annealing (SA) [19, 20], great deluge (GD) [21] and tabu search (TS) [22] require only one initial timetable in order to proceed with the improvement process. In the other hand, population-based methods used in solving CBCTT include ant colony optimization (ACO) [23], artificial bee colony (ABC) [15, 24], genetic algorithm (GA) [25, 26] and harmony search algorithm (HSA) [27, 28] require population of initial timetable in its improvement process. To produce a population of initial timetable requires algorithm that can produce multiple feasible timetables and these timetables must also be diverse. This process is a vital stage because it can influence the convergence speed i.e. process of decreasing the soft constraint violation and also the quality of the final timetable [29].

Hence, this study implemented an events assignment procedure that used different graph heuristics simultaneously in two phase approach. The phases consists of typical events assignment procedure in the first phase and also procedure for unassigned events in the second phase.

This study was able to construct population of initial timetable for various CBCTT data instances, therefore, generally, this study can contribute to the metaheuristic improvement approach. Specifically, this study can contribute to population-based methods of metaheuristic that use population of initial timetables such as ant colony optimization (ACO), artificial bee colony (ABC), genetic algorithm (GA), and harmony search algorithm (HSA).

2. RESEARCH METHOD

The proposed construction method is shown in Figure 1. The first step in the method is to determine the sequential order of courses/lectures to be scheduled using the combination of graph heuristics. From six different graph heuristics described in [11], this study investigates only three type of the heuristics. These heuristics have been selected because they are the most commonly used graph heuristics and have generated feasible initial timetables for all data instances of CBCTT [14, 16, 17]. The graph heuristics are LD, LW, and SD.

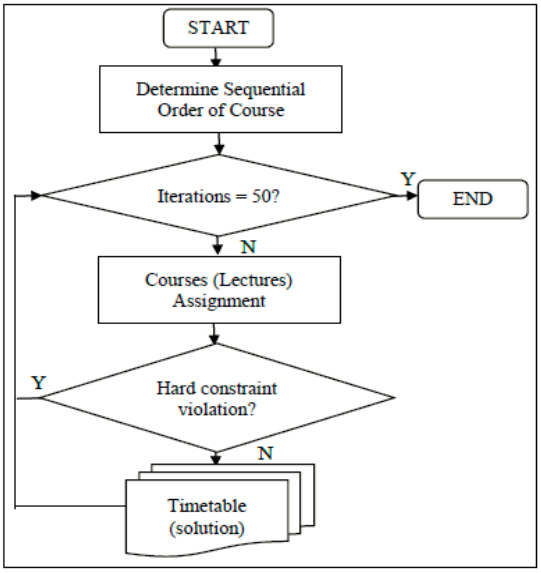

Figure 1. Approach for constructing population of initial solution

The courses were ordered using single heuristic, and a combination of two heuristics. The ordering method is identified by the following label of combination(s): L (LD), W (LW), S (SD), LS (LD with SD), WS (LW with SD), SL (SD with LD), and SW (SD with LW). LW is a heuristic that orders the events by the descending number of students involved in conflicts. This heuristic already contains the LD (descending number of conflicts) heuristic, therefore there is no combination between LD and LW.

In the second step, each of the lectures/courses which is previously arranged based on the heuristics setting will be randomly and iteratively allocated to valid empty slots while satisfying all the hard constraints. If a lecture is unable to be assigned to any slots because of no more valid empty slots, it will be inserted into the unscheduled lectures list. The unscheduled lectures/courses list will be assigned later to the timetable using several methods executed in a sequence.

The unassigned lecture assignment procedures consist of nine procedures. Each procedure tries to assign all the unassigned lectures to a valid timeslot. If there are more unassigned lectures, the next procedures will be executed. This implies that the current procedures are not able to assign some lectures; therefore, the next procedure will be attempted.

Procedure 1: Randomly assign the unassigned lecture to the available slot that is empty, free from conflict and conform to the unavailability constraints.

Procedure 2: Find an available slot that is empty and conforms to the unavailability constraints but have one lecture, which is in conflict with the unassigned lecture. As shown in Figure 2, the scheduled

lecture, 13, that contributed to the conflict, which is located in the same period (Slot i) with the available slot will be moved to another slot (Slot j) that is empty (the properties of suitable room, free from conflict, and conforming to the unavailability constraints are also maintained in this movement). This movement makes the available slot free from conflict and the unassigned lecture that is 11 can be scheduled in the available slot in Room 1.

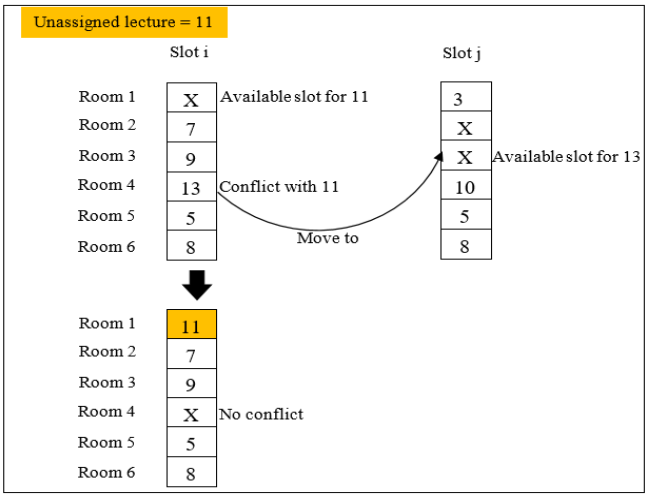

Figure 2. Graphic Illustration for Procedure 2

Procedure 3: Find an available slot that is not empty (there is a lecture already scheduled to it) but the slot is appropriate in terms of free from conflict and conforming to the unavailability constraints with the unassigned lecture. The scheduled lecture, i.e. 6, will be moved to another slot (Slot j) that is empty (the properties of suitable room, free from conflict, and conforming to the unavailability constraints are also maintained in this movement). The unassigned lecture that is 11 can then be scheduled in the slot (Slot i) that is empty. Figure 3 shows the graphic illustration for this step.

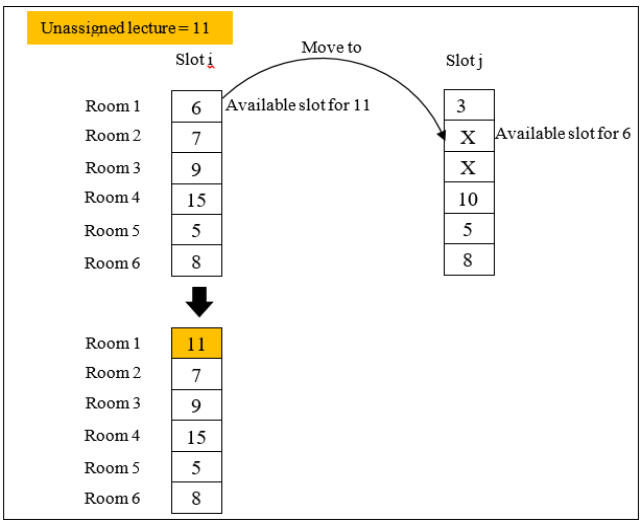

Figure 3. Graphic Illustration for Procedure 3

Procedure 4: Find an available slot that is not empty (there is a lecture already scheduled to it) and conforms to the unavailability constraints but have one conflict with the unassigned lecture. As shown in Figure 4, the scheduled lecture, 13 that contributes to the conflict which is located in the same period (Slot i)

with the available slot will be moved to another slot (Slot j) that is empty (the properties of suitable room, free from conflict, and conforming to the unavailability constraints are also maintained in this movement). This movement makes the available slot free from conflict. The next step is to move the scheduled lecture, 6 that is located in the available slot into the empty slot left by scheduled lecture, 13 that contributes to the conflict (the properties of suitable room, free from conflict, and conforming to the unavailability constraints are also maintained in this movement). Finally, the unassigned lecture 11 can be scheduled in the available slot.

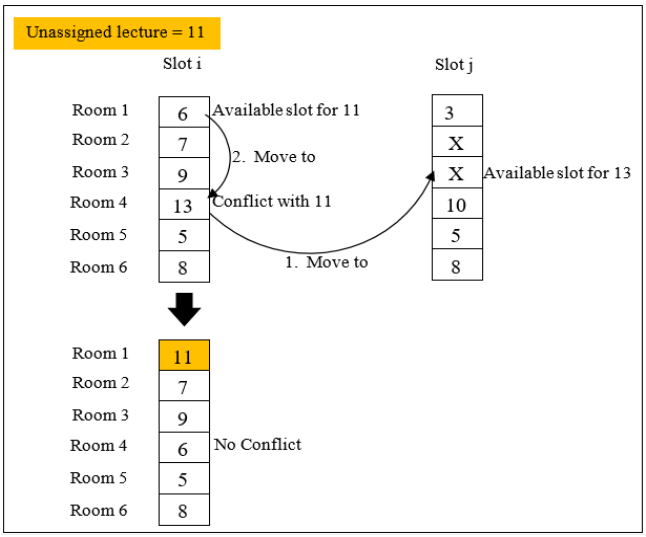

Figure 4. Graphic Illustration for Procedure 4

Procedure 5 until Procedure 8 will use the same procedure with Procedure 1 until Procedure 4 respectively. The difference with Procedure 5 to Procedure 8 is that the selection of available slot does not conform to unavailability constraints. This approach is used to provide alternative in the process of satisfying the hard constraints only since the unavailability constraints are considered to be the soft constraints. Procedures 5 to 8 relax the unavailability constraints while satisfying the hard constraints. In other words, the resulting timetable is still feasible but with higher number of soft constraints.

Procedure 9 is executed in which each scheduled slot in the timetable will be moved to an available empty slot (the properties of suitable room, free from conflict, and conforming to the unavailability constraints are also maintained in this movement). This step will change the overall timetable which contributes to provide different available slots in the next inner iteration.

Procedures 1 to 9 are repeated with different available slots (because Procedure 9 changes the overall slot in the timetable) from previous iterations. This procedure is iterated up to 20 times. This number of iterations is chosen because based on earlier experiments, normally, a feasible solution is found within 20 iterations.

After all the lectures are scheduled, the timetable will be validated. If the timetable is feasible, the algorithm stores the timetable and starts another courses (lectures) assignment procedure with another random seed. Otherwise, the present timetable will be eliminated and another courses (lectures) assignment procedure will be executed with a different random seed. The procedure of courses (lectures) assignment will be iterated for 50 times so that a maximum of 50 feasible timetables can be generated.

3. RESULTS AND ANALYSIS

The proposed construction approach was developed using C++ language and experimented using computer with an Intel Core i7 with 3.4 GHz processor. The experiment was carried out using six problem instances (comp01, comp02, comp03, comp04, comp05 and comp06) from International Timetabling Competition (ITC) 2007 available at http://tabu.diegm.uniud.it/ctt website. The proposed construction approach with several graph heuristic settings were performed for each problem instance, without imposing a time limit as a stopping condition. For each problem instance, the total number of feasible solutions of over 50 iterations will be produced.

The results of individual graph heuristic of L, W and S and combination of graph heuristics of LS, WS, SL and SW are shown in Tables 2 and 3 respectively. In the tables, TOT represents the total number of feasible initial solutions produced over 50 iterations, while MIN and MAX denote as the minimum cost and maximum cost of soft constraints violation that is produced.

Table 2. Results of Feasible Initial Solution of Single Heuristic

Problem	L			W			S		
Instances	TOT	MIN	MAX	TOT	MIN	MAX	TOT	MIN	MAX
comp01	50	346	646	50	365	900	50	272	525
comp02	50	781	1409	50	757	1586	50	728	1482
comp03	49	682	1193	50	705	1277	50	724	1157
comp04	50	708	843	50	696	987	50	702	871
comp05	2	1625	2027	3	1405	1769	2	1526	1999
comp06	50	957	1276	50	952	1595	50	961	1360

Table 3. Results of Feasible Initial Solution of Combination Heuristic

Problem	LS			WS			SL			SW		
Instances	TOT	MIN	MAX	TOT	MIN	MAX	TOT	MIN	MAX	TOT	MIN	MAX
comp01	50	330	569	50	326	690	50	323	489	50	366	559
comp02	50	769	1345	50	762	1424	50	747	1356	50	779	1377
comp03	50	702	881	50	701	1209	50	715	1129	50	690	1125
comp04	50	694	831	50	705	934	50	692	862	50	717	872
comp05	4	1466	1890	2	1313	1750	3	1594	2098	1	1296	-
comp06	50	947	1448	50	964	1473	50	982	1273	50	953	1368

Tables 2 and Table 3 show the results of the construction approach with all graph settings for all problem instances. Over 50 iterations, the construction approach is able to produce 50 feasible initial solution for all problem instances except for problem instances comp05. The highlighted row shows that the construction approach with all graph settings produce less then 10 percent of feasible initial solutions over 50 iterations for comp05. This is probably because of the complexity of the problem instance comp05.

Therefore, for the purpose of comparison, the construction approach will be executed with 10 runs on the problem instance of comp05 in order to investigate the best setting of graph heuristic used in the construction approach. Table 4 shows the number of feasible initial solutions of comp05 problem instance produced over 50 iterations for each graph heuristic setting with 10 runs.

Table 4. Number of Feasible Initial Solutions Produced over 50 Iterations for 10 runs on Comp05

Run	L	W	S	LS	WS	SL	SW
1	2	2	1	3	2	7	3
2	3	4	1	2	1	2	5
3	4	0	1	4	4	2	2
4	4	0	0	2	0	3	2
5	1	3	1	2	1	3	5
6	6	3	2	4	1	2	2
7	1	3	0	2	2	3	6
8	0	1	2	2	1	0	3
9	0	4	1	1	0	3	3
10	1	2	1	2	3	3	4
TOTAL	22	22	10	24	15	28	35

A statistical analysis, i.e. single factor analysis of variance (ANOVA) is carried out to inspect whether each setting has a significant difference in producing a number of initial solutions. A single factor ANOVA is used to test the null hypothesis, H_0 (H_0: The means of several groups are all equal). The hypothesis for this analysis, which is:

H_0: $\mu_L = \mu_S = \mu_W = \mu_{LS} = \mu_{WS} = \mu_{SL} = \mu_{SW}$

H_1: Not all the means are equal,

was performed at 5% significance level. The single factor ANOVA produced result called as p-value which consisted of probability number ranging from zero to one. The p-value is used to measure the difference in population means and used as an evidence to reject or accept the null hypothesis. Table 5 shows the result of ANOVA. The p-value for the hypothesis test is 0.007. This value is less than the specified significance level of 0.05, therefore H_0 is rejected. At 5% significance level, there is a significant difference between the graph heuristics settings in producing the feasible initial solutions.

Table 5. Result of ANOVA for Measuring the Different in the Graph Heuristics Settings

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	40.14286	6	6.690476	3.287832	0.007059	2.246408
Within Groups	128.2	63	2.034921			
Total	168.3429	69				

However, the ANOVA analysis does not show where the difference lies. Another analysis, i.e. t-test, is carried out to test the null hypothesis that the means of two populations are equal. In this case, each setting is paired and the t-test value of each pair is produced as shown in Table 6. At 5% significance level, null hypothesis, i.e. both paired settings have no significant difference, is rejected for W:S (0.03), W:WS (0.001), W:SW (0.01), S:LS (0.001), S:SL (0.02), S:SW (0.001), and WS:SW (0.01). In other words, these seven pairs of graph heuristic settings have significant difference in producing the initial solutions.

Table 6. Result of T-test for Each Pair of Graph Heuristic Settings

	L	W	S	LS	WS	SL	SW
L		1	0.09634	0.66178	0.297715	0.496102	0.201495
W			0.036787	0.764041	0.001005	0.404755	0.013276
S				0.001323	0.343436	0.023856	0.001617
LS					0.121232	0.533786	0.128623
WS						0.089781	0.01682
SL							0.343436
SW							

Based on the total number of feasible initial solutions produced as shown in Table 4 earlier and Table 6, the overall best setting(s) can be determined as listed in Table 7 by shortlisting the best settings in each pair. In pair 1 and pair 2, the graph heuristic W is determined as the best so far because it has greater total number of feasible initial solutions compared to graph heuristic S and WS. However, when compared to pair 3, graph heuristic W is no longer considered the best setting because graph heuristic SW has greater total number of feasible initial solutions compared to graph heuristic W. In pair 4 and pair 5, LS and SL are both better than their pair and each of them is included together with SW as the best graph heuristic settings so far. In pair 6 and pair 7, it seems that SW has better total number of feasible initial solutions compared to their pair. At the end, the best overall settings consist of SW, LS and SL.

Table 7. Shortlisting the Pairs of Graph Heuristic to Determine the Best Graph Heuristic Setting(s)

No	Pair of Settings	Total Number of Initial Solutions	Best setting in pair	Overall Best setting
1	W : S	22:10	W	W
2	W : WS	22:15	W	W
3	W : SW	22:35	SW	~~W~~, SW
4	S : LS	10:24	LS	SW, LS
5	S : SL	10:28	SL	SW, LS, SL
6	S : SW	10:35	SW	SW, LS, SL
7	WS : SW	15:35	SW	SW, LS, SL

To determine the best setting among SW, LS and SL, another 10 runs of comp05 using these settings are carried out. The total number of initial solutions produced over 50 iterations for 20 runs of SW, LS and SL is shown in Table 8. It is observed that graph heuristics SW and SL have the probability of producing zero (0) feasible initial solutions in Run 18 and 8 respectively. Based on this, it can be concluded that the graph heuristic LS is the best setting compared to SW and SL.

Table 8. Feasible Initial Solutions Produced by SW, LS and SL over 50 Iterations on Comp05

Run	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
SW	3	5	2	2	5	2	6	3	3	4	3	4	1	2	1	4	2	0	3	2
LS	3	2	4	2	2	4	2	2	1	2	3	4	3	2	4	5	3	3	2	3
SL	7	2	2	3	3	2	3	0	3	3	3	4	2	3	3	8	2	2	4	4

To prove the proposed construction approach with LS heuristic is the best setting, the proposed approach of LS heuristic setting is applied to other problem instances that are available at http://tabu.diegm.uniud.it/ctt website such as comp07, comp08, comp09, comp10, comp11, comp12, comp13, comp14, comp15, comp16, comp17, comp18, comp19, comp20 and comp21. Table 9 shows the results of applying proposed construction approach with LS heuristic for the other problem instances. The approach can produce maximum population of feasible initial solution for most of the problem instances.

Table 9. Results of Feasible Initial Solution of Construction Approach with LS Heuristic for Other Instances

Problem Instances	TOT	MIN	MAX
comp07	50	1043	1310
comp08	50	790	929
comp09	50	847	1064
comp10	50	898	1074
comp11	50	230	312
comp12	50	1498	2044
comp13	50	793	969
comp14	50	745	903
comp15	50	702	881
comp16	50	949	1117
comp17	48	902	1270
comp18	50	583	773
comp19	50	637	1225
comp20	50	1042	1282
comp21	50	928	1260

4. CONCLUSION

This paper portrays a construction approach that uses a combination of graph heuristics to construct a population of feasible initial timetables of curriculum based course timetabling problem. Results demonstrate that the construction approach with the use of largest degree followed by saturation degree setting is able to produce the maximum number of population instead of the use of single graph heuristics. The results of this study can contribute to the second phase of solving CBCTT problem that is the improvement stage, especially for population based metaheuristic methods.

ACKNOWLEDGEMENTS

The authors want to thank the Universiti Utara Malaysia for funding this study under the Research Generation University Grant Scheme, S/O code 13848 and RIMC, Universiti Utara Malaysia for the administration of this study.

REFERENCES

[1] A. Bonutti, *et al.*, "Benchmarking curriculum-based course timetabling: formulations, data formats, instances, validation, visualization, and results," *Ann. Oper. Res.*, pp. 1-12, 2010.
[2] L. Di Gaspero, *et al.*, "The Second International Timetabling Competition (ITC-2007): Curriculum-based course timetabling (track 3)," *School of Electronics, Electrical Engineering and Computer Science, Queens University, Belfast (UK). (Technical Report QUB/IEEE/Tech/ITC2007/CurriculumCTT/v1.0/1)*, 2007.
[3] S. Petrovic, "Towards the Benchmarks for Scheduling," in *The International Conference on Automated Planning and Scheduling, ICAPS07*, 2007.
[4] D. Wood, "A System for Computing University Examination Timetables," *Comput. J.*, vol. 11, pp. 41-47, 1968.
[5] J. Brittan and M. Farley, "College Timetable Construction by Computer.," *Comput. J.*, vol. 14, pp. 361-365, 1971.
[6] L. Gilbert and D. Sylvain, "Examination timetabling by computer," *Comput. Oper. Res.*, vol. 11, pp. 351-360, 1984.
[7] W. Carter, *et al.*, "Examination timetabling: Algorithmic strategies and applications," *J. Oper. Res. Soc.*, vol. 47, pp. 373-383, 1996.
[8] H. Asmuni, *et al.*, "Fuzzy Multiple Ordering Criteria for Examination Timetabling, In E. Burke, M. Trick (Eds.),"

in *The 5th International Conference of Practice and Theory of Automated Timetabling, 18th - 20th August 2004, Pittsburgh, PA USA*, 2004.
[9] S. A. Rahman, *et al.*, "A nonlinear heuristic modifier for constructing examination timetable," *J. Theor. Appl. Inf. Technol.*, vol. 95, pp. 5642-5653, 2017.
[10] S. A. Rahman, *et al.*, "Graph coloring heuristics for solving examination timetabling problem at Universiti Utara Malaysia," *AIP Conf. Proc.*, vol. 1635, pp. 491-496, 2014.
[11] E. K. Burke, *et al.*, "A graph-based hyper-heuristic for educational timetabling problems," *Eur. J. Oper. Res.*, vol. 176, pp. 177-192, 2007.
[12] Z. Lu and J. K. Hao, "Adaptive Tabu Search for course timetabling," *Eur. J. Oper. Res.*, vol. 200, pp. 235-244, Jan 2010.
[13] M. J. Geiger, "Applying the threshold accepting metaheuristic to curriculum based course timetabling," *Ann. Oper. Res.*, vol. 194, pp. 189-202, 2010.
[14] A. L. Bolaji, *et al.*, "Artificial Bee Colony Algorithm for Curriculum-Based Course Timetabling Problem," in *The 5th International Conference on Information Technology (ICIT) 2011, May 11-13, 2011, Amman Jordan*, 2011.
[15] A. L. Bolaji, *et al.*, "An Improved Artificial Bee Colony for Course Timetabling," in *Sixth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), 2011*, pp. 9-14, 2011.
[16] K. Shaker and S. Abdullah, "Incorporating great deluge approach with kempe chain neighbourhood structure for curriculum-based course timetabling problems," in *2nd Conference on Data Mining and Optimization, 2009. DMO '09. 27-28 Oct. 2009*, pp. 149-153, 2009.
[17] S. Abdullah, *et al.*, "An Investigation of a Genetic Algorithm and Sequential Local Search Approach for Curriculum-based Course Timetabling Problems," *Multidiscip. Int. Conf. Sched. Theory Appl. (MISTA 2009) 10-12 August 2009, Dublin, Irel.*, 2009.
[18] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization:Overview and conceptual comparison," *ACM Comput. Surv.*, vol. 35, pp. 268-308, 2003.
[19] H. Y. Tarawneh, *et al.*, "A Hybrid Simulated Annealing with Solutions Memory for Curriculum-based Course Timetabling Problem," *J. Appl. Sci.*, vol. 13, pp. 262-269, 2013.
[20] R. Bellio, *et al.*, "A simulated annealing approach to the curriculum-based course timetabling problem," *6th Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA 2013)*, 2013.
[21] J. Wahid and N. M. Hussin, "Hybrid harmony search with great deluge for UUM CAS curriculum based course timetabling," *J. Telecommun. Electron. Comput. Eng.*, vol. 9, pp. 33-38, 2017.
[22] S. Abdullah and H. Turabieh, "On the use of multi neighbourhood structures within a Tabu-based memetic approach to university timetabling problems," *Inf. Sci. (Ny).*, vol. 191, pp. 146-168, May 2012.
[23] K. Socha, *et al.*, "A max-min ant system for the university course timetabling problem," *Proc. 3rd Int. Work. Ant Algorithms (ANTS 2002).Lecture Notes Comput. Sci. Vol. 2463. Springer-Verlag, Berlin*, pp. 1-13, 2002.
[24] S. Agahian, *et al.*, "Adaptation and Use of Artificial Bee Colony Algorithm to Solve Curriculum-based Course Time-Tabling Problem," *Fifth Int. Conf. Intell. Syst. Model. Simul.*, pp. 78-82, 2014.
[25] C. Akkan and A. Gülcü, "A bi-criteria hybrid Genetic Algorithm with robustness objective for the course timetabling problem," *Comput. Oper. Res.*, vol. 90, pp. 22-32, 2018.
[26] R. Lewis and B. Paechter, "Application of the Grouping Genetic Algorithm to University Course Timetabling," in G. Raidl and J. Gottlieb, "Evolutionary Computation in Combinatorial Optimization," Springer Berlin, Heidelberg, vol. 3448, pp. 144-153, 2005.
[27] J. Wahid and N. M. Hussin, "Solving Curriculum Based Course Timetabling by Hybridizing Local Search Based Method within Harmony Search Algorithm," in M. Berry, *et al.*, "Soft Computing in Data Science SE - 14," Springer Singapore, vol. 545, pp. 141-153, 2015.
[28] M. A. Al-Betar and A. T. Khader, "A harmony search algorithm for university course timetabling," *Ann. Oper. Res.*, vol. 194, pp. 1-29, 2010.
[29] S. Rahnamayan, *et al.*, "A novel population initialization method for accelerating evolutionary algorithms," *Comput. Math. with Appl.*, vol. 53, pp. 1605-1614, 2007.

BIOGRAPHIES OF AUTHORS

Juliana Wahid received her PhD in Technology Information at the Universiti Teknologi MARA of Malaysia in 2017. Her PhD thesis is on Solving University Course Timetabling. She is a senior lecturer in the School of Computing at the Universiti Utara Malaysia. She is also working as an associate fellow under the Institute of Advanced and Smart Digital Opportunities (IASDO). Her area of interest includes computational optimization, scheduling, timetabling, heuristics and metaheuristics. She has published her works in several journal including Journal of Soft Computing Software Engineering, Journal of Telecommunication, Electronic and Computer Engineering, and Journal of Information and Communication Technology.

Syariza Abdul-Rahman is an associate professor of Decision Science in the School of Quantitative Sciences at the Universiti Utara Malaysia. She is also working as an assistance research fellow under the Institute of Strategic Industrial Decision Modeling (ISIDM). Her interest includes operational research, computational optimization, scheduling, timetabling and heuristics and metaheuristics. She received her PhD in Computer Science at the University of Nottingham in 2012. Her PhD thesis is on Search Methodologies for Examination Timetabling. She has published her works in European Journal of Operational Research, Annals of Operations Research, Journal of Information and Communication Technology, Advances in Operations Research and many more.

Aniza Mohamed Din holds a Bachelor in Information Technology from Universiti Utara Malaysia in 1998 and a Masters degree in Computer Science from Universiti Sains Malaysia in 1999. Her research interests include resource allocation problem, optimization, metaheuristics, genetic algorithm and artificial immune systems.

Naimah Mohd-Hussin is an associate professor in the Faculty of Computer and Mathematical Sciences at the Universiti Teknologi MARA, Perlis, Malaysia. Her interest includes optimization algorithm, scheduling and timetabling. She received her PhD in Computer Science at the University of Nottingham in 2005. Her PhD thesis is on Solving Examination Timetabling. She has published her works in Springer Lecture Notes, Journal of Telecommunication, Electronic and Computer Engineering, Journal of Soft Computing Software Engineering and many more.